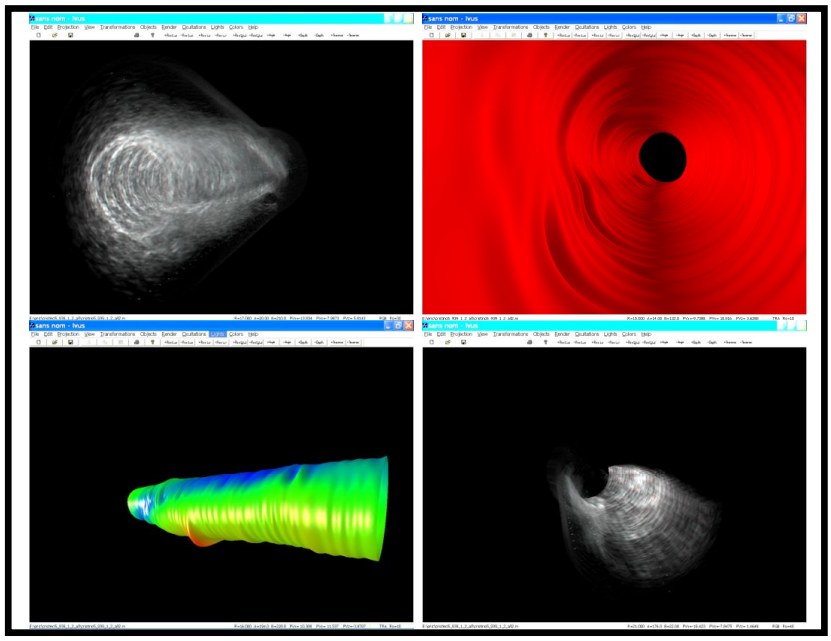
UNIVERSITE PAUL SABATIER TOULOUSE III

IUP Systèmes Intelligents

IUP SI M1

VISUALISATION 3D DE VAISSEAUX SANGUINS

Rapport de Stage



IUP SI M1

AMIEL Eric

TABLE DES MATIERES

INTRODUCTION

Aujourd'hui, les angiogrammes représentent le principal diagnostic clinique de vaisseaux sanguins. Cependant, la nature subjective des images X-ray, des angiogrammes rend la visualisation très limitée et des informations telles que la distribution ou encore l'extension des maladies sur les parois du vaisseau ne peuvent être définies. En complément, les images Intra Vascular Ultrasound (IVUS) représentent pour les physiciens, l'unique visualisation permettant d'avoir une photo de la composition d'un vaisseau en détail. Les IVUS contiennent des informations géométriques valides (diamètre, aire etc.) concernant par exemple les plaques (plaques de calcium se fixant sur les bords du vaisseau, cause de nombreuses maladies), lumen (ce qui constitue « grossièrement » l'intérieur du vaisseau) ou encore la position d'un stent à l'intérieur du vaisseau.

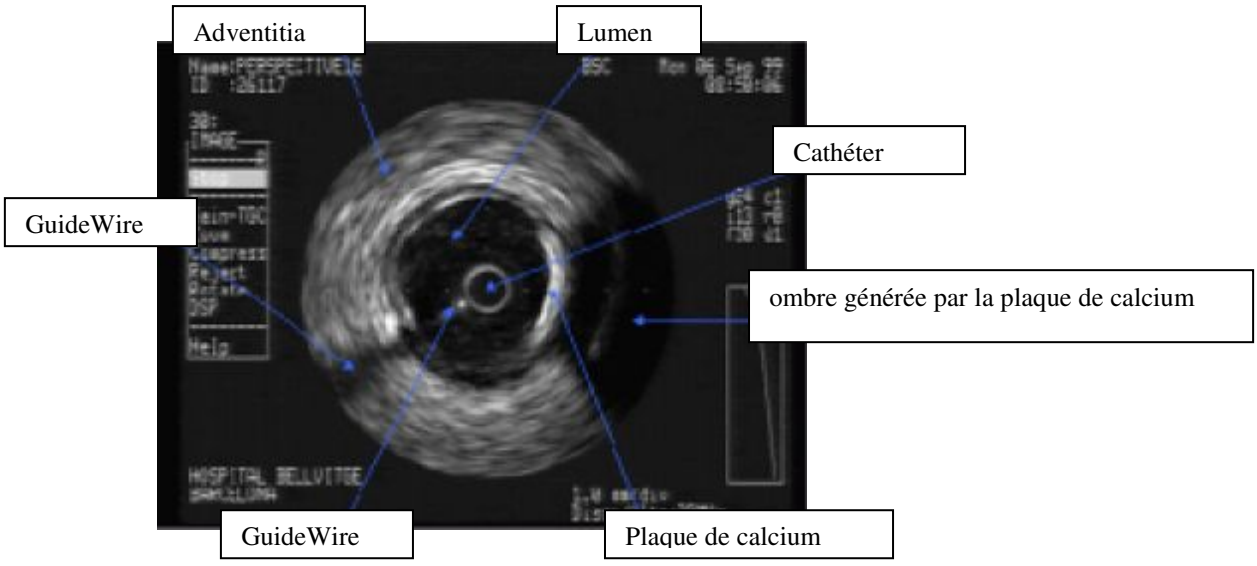
Mon travail consistait donc à réaliser un programme permettant de faciliter la lourde tâche mentale de conceptualisation de la forme de la coronaire et d'analyse de réelles extensions, distributions spatiales de la maladie du vaisseau (coronaire par exemple).

Effet, le but de mon stage au sein du Laboratoire CVC (Computer Vision Center) était de réaliser un programme interactif de modélisation/visualisation 3D de vaisseaux (veines /artères) à partir de données issues d'un traitement préliminaire sur des images IVUS (Intra Vascular Ultrason System). J'ai intégré une équipe qui travaille sur la segmentation des contours d'images IVUS et utilisé leurs travaux pour modéliser les vaisseaux en trois dimensions.

Principe des IVUS : un cathéter avec un capteur à son extrémité est introduit à l' intérieur de la coronaire. Le capteur IVUS peut faire des rotations, envoie des pulsations ultrasons. Ce procédé permet la réalisation d'une vidéo sur une console.

Exemple d'une image IVUS :

Figure 1



Exemple IVUS

Dans la figure 1, l' IVUS représente une coupe transversale d'un vaisseau ou différentes structures morphologiques peuvent être observées.

Le cathéter et la courbe caractérisant son trajet à l'intérieur du vaisseau, les plaques de calcium, Adventitia ou encore les parois des vaisseaux sont les éléments principaux sur lesquels s'est appuyés mon projet.

PRESENTATION DU LABORATOIRE

Le centre de vision par ordinateur (Computer Vision Center) est un centre consacré à la recherche et au développement dans le domaine de la vision par ordinateur. Il a été établi en 1995 par Industry Department and the CIRIT ("National Science Foundation") of the Generalitat de Catalunya and the Universitat Autònoma de Barcelona (UAB).

L'objectif stratégique de CVC est de faire à la fois de la recherche et du développement de bonne qualité dans le domaine de la vision par ordinateur.

D'un point de vue scientifique, le CVC souhaite élargir les connaissances dans ce domaine. D'un point de vue technologique, les objectifs CVC consistent à contribuer à l'innovation ainsi qu'à la compétitivité industrielle en collaborant avec des sociétés de haut niveau dans le but de réaliser des projets technologiques.

UTILISATION DE L'ENVIRONNEMENT MFC

Nous avons choisi d'implémenter notre programme sous le système d'exploitation Windows afin de pouvoir profiter de l'environnement MFC développé par Microsoft Visual Studio. En effet, nous souhaitions réaliser un programme interactif (Menu, interactions avec divers boutons,clavier, souris etc.) et cet environnement propose une fenêtre,une toolBar pré implémentés. De plus, OpenGL fut facile a intégré dans cet environnement puisque MfC est adapté au langage C++.

Hormis la prise en main de l'environnement MFC de Microsoft Visual Studio, mon premier travail fut d'implémenter une interface de programme permettant de manipuler des objets modélisés en OpenGL dans un espace 3D. En effet, n'importe quel objet rendu dans la fenêtre de visualisation devait pouvoir être transformé par translation, rotation, scalage. Pour améliorer la future visualisation du vaisseau, nous avons implémenté certaines options comme :

- Modification de la couleur du fond de la fenêtre de visualisation afin de plus ou moins faire ressortir certaines couleurs du vaisseau par contraste.
- Activation/désactivation des parties cachées de l'objet.
- Modification du type de lumière (suave, plane, filaire, fixe par rapport à la caméra).

RENDU REALISTE DU VAISSEAU

Le rendu réaliste du vaisseau s'est décomposé en 2 parties. La première consistait à définir un maillage de points, la deuxième à définir la surface s'appuyant sur le maillage précédemment défini.

1. *Définition du maillage*

L'hôpital nous a fourni des images IVUS d'un patient ainsi que les positions 3D du cathéter pour chacune de ces images.

L'équipe dans laquelle je fus intégré travaillait sur la segmentation des contours d'images IVUS et a constitué un fichier réunissant les coordonnées des points de contour pour chacune de ces images (appelées aussi « coupes »).

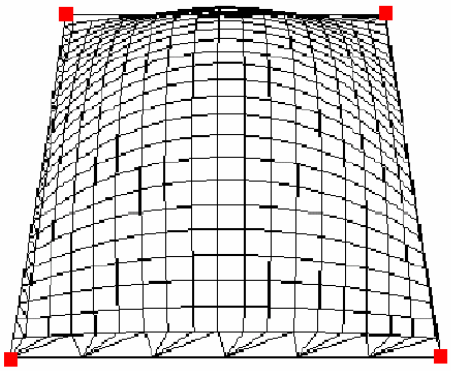
Au départ, mon travail a consisté, à partir des images communiquées, du fichier résultat de l'équipe et de part la position 3D du cathéter dans chaque image, de réaliser un maillage de points dans l'espace en trois dimensions.

2. *Organisation du maillage*

La technique de rendu retenu à été la technique des NURBS. Cette technique permet d'obtenir des surfaces lisses à partir de quelques points au lieux d'avoir une surface présentant des arrêtes (dans le cas d'un rendu par la technique des polygones).

Les NURBS sont des surfaces que l'on pourrait qualifiées carrées. C'est à dire qu'elles s'appuient sur obligatoirement 4 points et il est alors impossible de représenter une surface fermée (comme celle d'un vaisseau semblable à un cylindre).Il est donc nécessaire de décomposer la surface « fermée » de notre vaisseau par plusieurs sous surfaces ouvertes représentables par des NURBS.

Figure 4



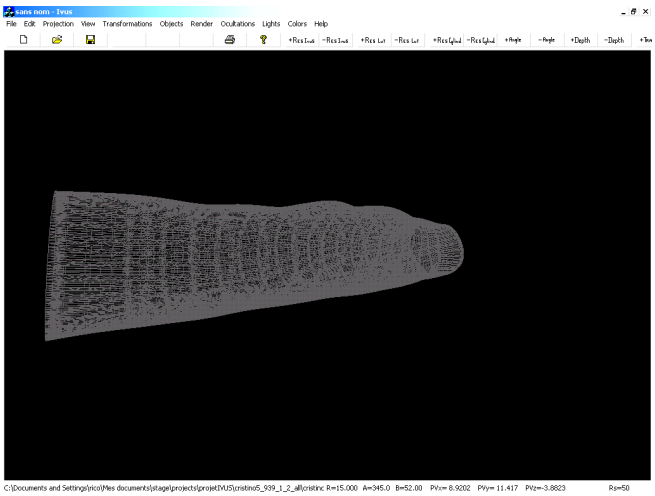
Surface NURBS

Ainsi,pour représenter ma surface fermée (vaisseau), j'ai fait le choix de modifier mon maillage de points. En effet, pour chaque coupe (image), aux 30 points initialement définis, j 'y ai ajouté les 4 premiers points de ces 30 points. En conséquence, la dernière sous surface ouverte a permis de fermer ma surface pour chaque coupe.

La principale contrainte de ce projet résidait dans le fait que le maillage devait être constitué du plus grand nombre de points possibles, afin d'avoir le rendu le plus réaliste possible, tout en permettant d'appliquer diverses opérations sur l'objet (lumière, couleurs,texture, transformations géométrique, etc.) et d'offrir la possibilité d'un parcours à l'intérieur du vaisseau assez fluide. Par exemple, je travaillais sur un patient dont la coronarographie était composée de 934 images, l'équipe avait fait le choix de définir le contour du vaisseau, pour chacune des images, par 30 points. Ainsi, mon objet 3D aurait été composé de 934*30 soit 28020 points. Rendre un objet composé de plus de 28000 points est possible pour toutes cartes graphiques 3D relativement récentes cependant,toutes opérations appliquées sur cet objet restent trop lourdes,le temps de calcul nécessaire beaucoup trop long.

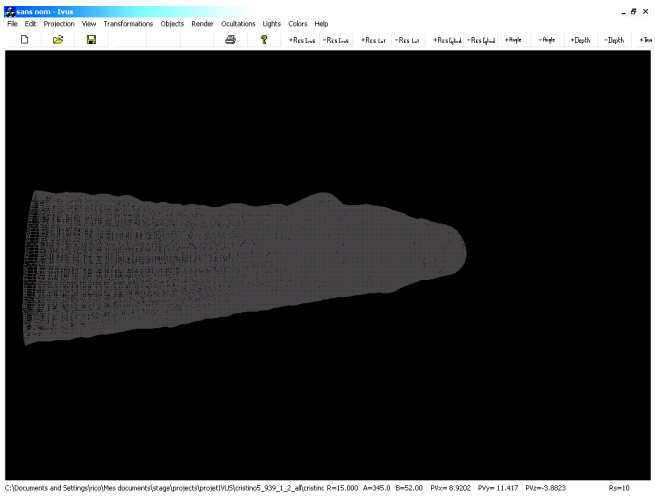
C'est pourquoi, j'ai proposé un mode « multi résolution » permettant de réduire/augmenter la résolution du rendu et ainsi adapté mon programme selon la puissance de la carte graphique utilisée. Ce mode « multi résolution » consiste à prendre en compte les points contours d'une image sur i, avec i Є [10,15,20,25,…,45,50].

Figure 2



Rendu 1/50 images

Figure 3



Rendu 1/10 images

Sur la figure 2, l'objet a été modélisé à partir des points contours issus d'1/50 images tandis que sur la figure 3, il a été modélisé à partir des points contours issus d'1/10 images. En effet, les détails (irrégularités sur le contour) sont plus importants sur la figure 3.

3. Problème(s) rencontré(s)

Mon principal problème résidait dans le manque d'information, documentation, concernant l'implémentation de NURBS à surface fermée.

Le second problème rencontré concernait la position initiale du vaisseau dans la fenêtre de visualisation. Une fois le vaisseau rendu, l'application d'une transformation quelconque (rotation, translation,scalage) sur l'objet ne produisait pas le résultat souhaité. En effet, ces transformations s'effectuent par rapport au repère de projection. Cependant le vaisseau rendu n'était pas, initialement, centré à l'origine de ce repère ni orienté selon ces axes. Ainsi, lorsqu'une transformation était appliquée, une erreur de position était induite.
J'ai donc fait le choix d'appliquer une translation,T,du centre du vaisseau vers l'origine du repère de projection, sur l'objet 3D (le vaisseau).J'ai aussi appliqué trois rotations {Rx,Ry,Rz} sur le vaisseau afin qu'il soit réorienté par rapport au repère de projection.

Le dernier problème consistait à modifier les coordonnées de chacune des positions du cathéter. En effet, les points composant la Spline du cathéter ne sont pas considérés comme des Vertex (Point dans l'espace 3D).Il était donc impossible d'utiliser les primitives OpenGL, j'ai ainsi défini une matrice de transformation correspondant à la translation T et aux rotations {Rx,Ry,Rz} que j'ai appliquée sur les coordonnées des points composants cette Spline.

PARCOURS ANIME A L'INTERIEUR DU VAISSEAU

1. *Présentation*

Le parcours animé à l'intérieur du vaisseau est un mode permettant à l'utilisateur de parcourir l'intérieur du vaisseau en suivant le trajet réalisé par le cathéter lors de la coronarographie. J'ai proposé 2 modes de parcours différents :

- Le premier propose à l'utilisateur de parcourir manuellement l'intérieur du vaisseau. En effet, il peut à l'aide du clavier, faire avancer/reculer le cathéter (la caméra en réalité), choisir de regarder vers l'avant/l'arrière du parcours. Il peut aussi, à l'aide de la souris, regarder partout autour de lui (modifier la position du point de vue de la caméra dans la fenêtre de visualisation).
- Le second propose à l'utilisateur un parcours automatique à l'intérieur du vaisseau à 2 vitesses différentes. Cette animation, peut être mise en pause en pressant la touche « espace ».

Rmq : Il y a 2 vitesses possibles puisque la vitesse du cathéter lors d'une coronarographie peut être de 0.5 m/s ou de 1.0 m/s.

Ce mode permet de rendre compte des problèmes possibles lors de l'insertion du cathéter à l'intérieur du vaisseau tout au long du parcours. Les irrégularités des bords du vaisseau (reliefs) sont mises en évidence de part la direction tangentielle du vecteur de visée de la caméra.

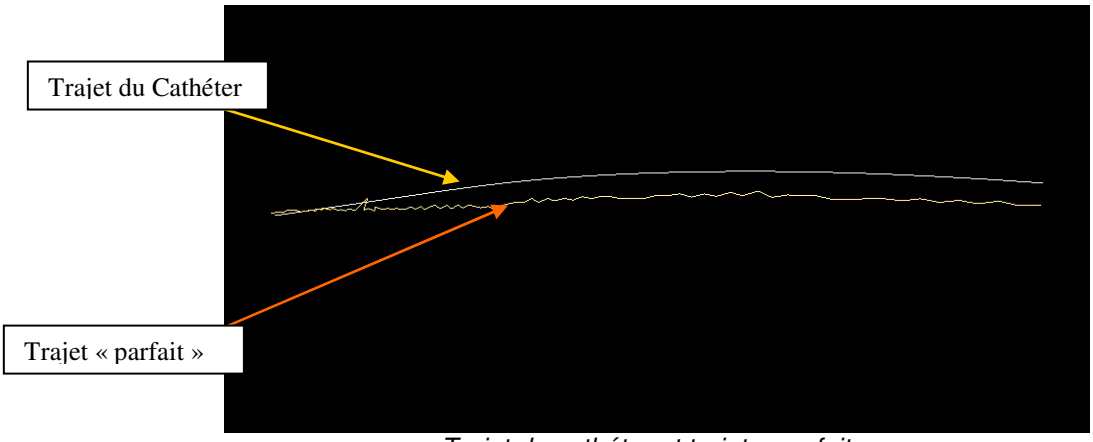
Nous avons choisi de faire apparaître une deuxième courbe pouvant constituer le trajet « parfait » (suivant l'axe central du vaisseau) du cathéter. Cette courbe permet de rendre compte de l'excentricité de certaines positions du cathéter et donc des approximations lors de la propagation du cathéter.

2. *Mode opératoire*

Tout d'abord, j'ai constitué la courbe retraçant le trajet du cathéter. Pour cela, j'ai choisi la technique des Splines. Cette technique permet d'obtenir une courbe lisse à partir des positions du cathéter pour chacune des coupes. Le mouvement de la caméra le long de la courbe est ainsi plus régulier. J'ai aussi calculé la DSpline afin de connaître les vecteurs normaux de chaque point de la Spline.

Ensuite, j'ai construit la courbe constituant le trajet « parfait », suivant l'axe central du vaisseau,que devrait suivre le cathéter. Cette courbe est composée des centres de gravités de chacune des coupes. Un centre de gravité, pour une coupe donnée, est calculé à partir des 30 points constituants le contour de cette même coupe.

Figure 4



Trajet du cathéter et trajet « parfait »

Sur la figure 4,nous pouvons voir la spline correspondant au trajet du cathéter ainsi que la courbe (« trajet parfait » correspondant aussi à l'axe central du vaisseau) composée des centres de gravité de chaque coupe.

Pour finir,il a fallu implémenter l'animation. Pour cela, j'ai tout d'abord défini un timer pré implémenté dans l'environnement MFC de Microsoft Visual Studio. A chaque signal délivré par le timer, les nouveaux paramètres de la caméra (position,point de vue,vecteur normal) sont calculés et la matrice de vision d'OpenGL est mise à jour.

Rmq :

- *le signal du timer est délivré automatiquement et périodiquement si le mode choisi par l'utilisateur est l'automatique. Dans le cas où l'utilisateur choisi le mode manuel,le signal est délivré lorsque l'utilisateur presse la touche permettant d'avancer/reculer,sur le clavier.*
- *La matrice de vision d'OpenGL permet de modifier la position et l'orientation de la caméra virtuelle. Pour ma part, j'ai choisi d'utiliser la procédure gluLookAt(…) pour spécifier cette matrice de vision. Cette procédure prend en paramètre la position de la caméra, la position du point visé, le vecteur normal de la caméra (définissant l'orientation de celle-ci).*

A chaque signal délivré par le timer :

La position de la caméra est mise à jour,elle correspond en fait à la position suivante (précédente,selon le sens du parcours) sur la Spline définissant le trajet du cathéter.

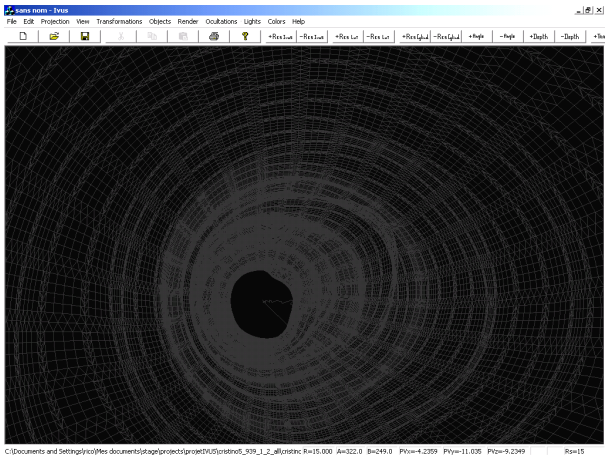
Les coordonnées du point visé sont mises à jour. Elles correspondent aux coordonnées du point suivant (précédent, selon le sens de vue) la position courante de la caméra sur la spline. Si l'utilisateur modifie la position du point de vue avec la souris, on calculera le nouveau point visé en ajoutant la variation induite au mouvement de la souris aux coordonnées du ce point.

Le vecteur normal de la caméra est mis à jour. Il y a 2 possibilités lors de la mise à jour de ce vecteur :

* Si le mode sélectionné est le mode automatique, les vecteurs normaux successifs correspondent aux vecteurs normaux du cathéter,pour chaque coupe,livrés par l'hospital (au cours de la coronarographie,le cathéter peut pivoter sur lui-même,l'orientation est défini par ce vecteur normal).

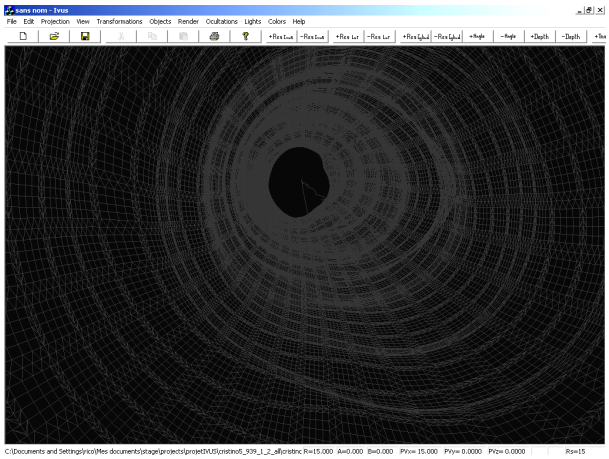
* Si le mode sélectionné est le mode manuel,le vecteur normal correspond au point suivant/précédent sur la DSpline (selon le sens de parcours). Si l'utilisateur modifie la position du point de vue avec la souris, on calculera le nouveau vecteur normal en ajoutant la correspondance sur l'axe vertical de la variation induite au mouvement de la souris aux coordonnées de ce vecteur.

Figure 5



Animation à un instant t

Figure 6



Animation à un instant t,avec pt visé modifié

On peut voir sur la figure 5 et 6, le parcours du vaisseau à un instant, t, donné. Dans la figure 5, le point de visé correspond au point suivant la position du cathéter sur la Spline, tandis que dans la figure 6, l'utilisateur a modifié la position de ce point visé avec la souris.

SPECIFICATION DES COULEURS ET DE L'ECLAIRAGE DU VAISSEAU

Deux objectifs étés fixés pour cette partie du projet. Le premier consistait à rendre le vaisseau en lui appliquant une couleur unique proposant un rendu réaliste du vaisseau. Le deuxième concernait un rendu du vaisseau en lui appliquant des couleurs caractérisées par une Lut (modifiable par l'utilisateur) faisant apparaître certaines caractéristiques morphologiques du vaisseau.

1. Couleur pour rendu réaliste

La première étape consistait à choisir la couleur la plus proche du sang. J'ai utilisé un logiciel de photo pour récupérer les composantes couleur sur une photo faisant apparaître du sang. Une fois cette couleur appliquée sur l'objet (le vaisseau) et le rendu ne faisant pas apparaître les irrégularités de la parois du vaisseau,nous avons décidé d'utiliser des lumières,autres que la lumière ambiante prédéfinie par OpenGL,et de définir des propriétés matérielles au vaisseau modélisé.
(voir figure 8)

2. Eclairage

Dans le modèle OpenGL, la couleur que l'on perçoit d'un objet dépend de la couleur propre de l'objet, de ses propriétés matérielles et de la couleur de la lumière qu'il reçoit.

J'ai donc définit deux sources de lumières, chacune positionnée aux extrémités du vaisseau en direction des extrémités opposées. Ces deux sources de lumières sont de types diffuses. Ce sont des lumières qui viennent d'une direction particulière et qui vont être plus brillantes si elles arrivent perpendiculairement à la surface que si elles sont rasantes. Par contre, après avoir rencontrés la surface, elles sont renvoyées uniformément dans toutes les directions. Ces deux lumières permettent de faire ressortir les irrégularités de la paroi du vaisseau puisque elles sont rasantes. *(voir figure 7 & 8)*

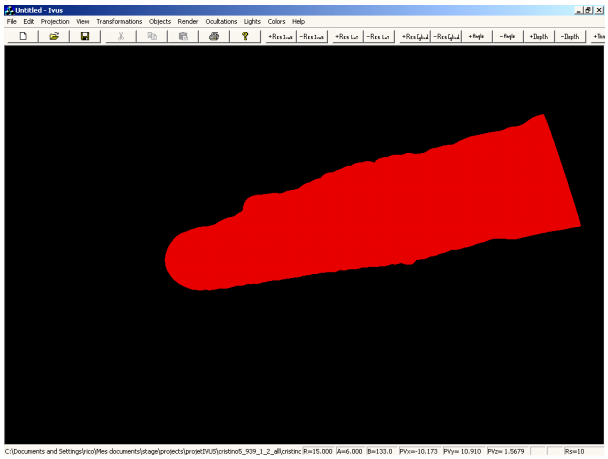
J'ai conservé la lumière ambiante prédéfinie par OpenGL. Cette lumière semble venir de toutes les directions, elle permet d'éclairer de manière suave les zones non éclairées par les deux lumières diffuses.

Lorsque le mode « parcours du vaisseau » est activé, je propose à l'utilisateur de modifier la position et la direction des deux lumières diffuses. La première pourrait être comparée à une « lampe frontale » puisqu'elle est placée sur la caméra et point vers le point de vue de la caméra. La deuxième est placée sur le point de vue en direction de la caméra. Cela permet de faire apparaître localement les irrégularités de la paroi du vaisseau lorsque le point de vue est modifié. *(voir figure 9)*

Pour rendre le rendu plus réaliste, j'ai fait en sorte qu'OpenGL considère pour ces calculs d'éclairage que l'œil est dans la scène que l'angle entre le point de vue et l'objet est pris en compte. Ainsi, lorsque l'utilisateur place la caméra de telle sorte que le vecteur de visée soit orthogonal à la surface du vaisseau (angle d'incidence nul), les irrégularités sur la surface du vaisseau apparaissaient moins.

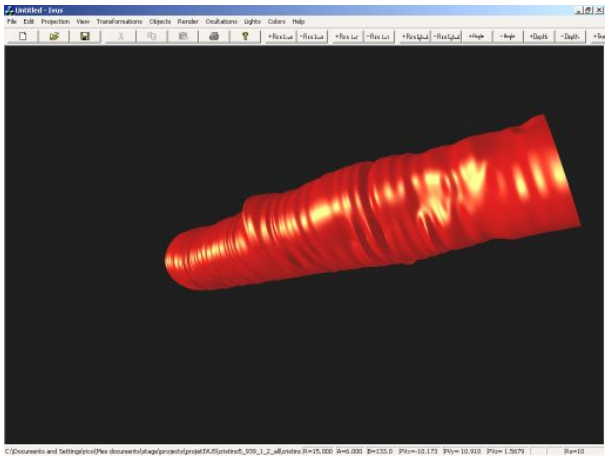
Aussi j'ai déterminé des propriétés matérielles à l'objet. En effet, j'ai définit des caractéristiques de spécularité et de brillance au vaisseau. La composante spéculaire de l'objet détermine une lumière renvoyée dans une direction particulière. La brillance détermine l'intensité et la taille de la tache de réflexion spéculaire. Ainsi, lorsque l'angle d'incidence est nul, les irrégularités sont plus apparentes grâce aux taches issues de la brillance.

Figure 7



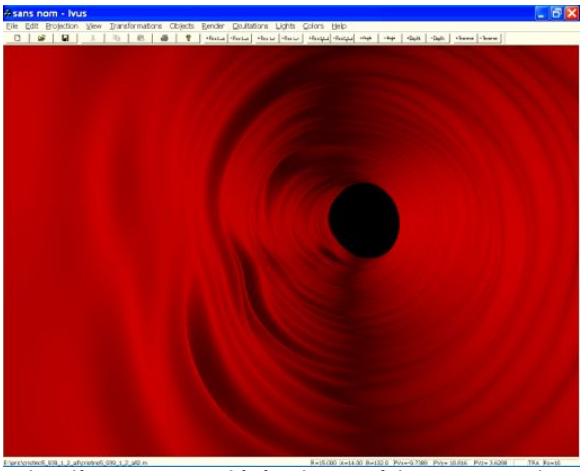
Rendu réaliste avec lumière ambiante seulement

Figure 8



Rendu avec lumières et propriétés de matériaux

Figure 9



Rendu avec lumières et propriétés de matériaux en mode « parcours »

Nous pouvons voir sur les figures 7 & 8 l'importance de l'éclairage de la scène ainsi que celle des propriétés de matériau du vaisseau faisant apparaître des taches brillantes sur la figure 8.

La figure 9 montre le résultat de l'éclairage en mode « parcours ».

3. *Couleurs pour détermination de caractéristiques morphologiques*

Le but, ici, était de faire ressortir certaines caractéristiques locales du vaisseau tel que son élargissement,ou encore les distances par rapport au cathéter ou par rapport au centre de gravité pour chacune des coupes. Toutes ces caractéristiques peuvent être le signe de maladie chez le patient.

Définition de la LUT :

Une fois définie, la même LUT sera utilisée pour définir les représentations du niveau d'élargissement, des distances par rapport au cathéter ou par rapport au centre de gravité pour chacune des coupes du vaisseau.

J'ai décidé définir la LUT à partir d'une base de trois couleurs, que l'utilisateur peut modifier en modifiant un fichier texte définissant cette base de couleurs.

La base par défaut est composée de trois positions principales auxquelles correspondent trois couleurs.

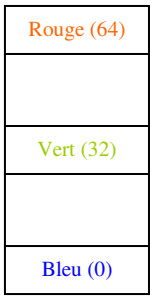
Par défaut :

La position la plus élevée dans la LUT est associée au Rouge.

La position moyenne dans la LUT est associée au Vert.

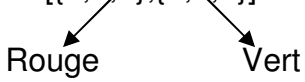
La position la plus faible dans la LUT est associée au Bleu.

Le nombre de couleurs (résolution) définissant la LUT est de 64. L'utilisateur peut modifier la résolution de la LUT.



Les couleurs correspondantes aux positions situées entre deux des trois paliers sont interpolées par une interpolation tri-linéaire entre ces deux paliers (interpolation linéaire sur chacune des trois composantes couleurs).

Si par exemple la couleur souhaitée est positionnée entre la position du Rouge et celle du Vert,l'interpolation tri-linéaire se fera dans l'intervalle [{1,0,0};{0,1,0}].



Elargissement du vaisseau :

Le principe de cette représentation couleur consiste à définir, pour chaque coupe,la couleur représentative de l'élargissement du vaisseau au niveau de cette coupe et assigner cette couleur à tous les points du vaisseau correspondant à cette coupe. Cette représentation a pour but de représenter les différents niveaux d'élargissement du vaisseau. En effet, une zone anormalement large ou étroite sera mise en évidence par une couleur représentative de l'élargissement caractéristique à cette zone.

L'algorithme de ce calcul est :

Variables utilisées :

nb_Coupe = entier égal au nombre de coupes composant le vaisseau.
largeur_moyenne[i] = distance euclidienne moyenne pour une coupe,i, entre les points de la coupe et le centre de gravité
largeur_moyenne_globale=valeur moyenne des moyennes de chaque coupe
largeur_moyenne_minimale = largeur moyenne minimale calculée parmi toutes les largeurs moyennes calculées.
Largeur_moyenne_maximale = largeur moyenne maximale calculée parmi toutes les largeurs moyennescalculées.
resol_LUT = résolution de la LUT (nombre maximum de couleurs définissant la LUT)
positionLUT = position de la couleur courante dans la LUT
LUT = matrice 3*3 contenant la couleur des trois paliers définissant la base de la LUT
Par défaut :

LUT[0] = 1.	LUT[1] = 0.	LUT[2] = 0.	Rouge
LUT[3] = 0.	LUT[4] = 1.	LUT[5] = 0.	Vert
LUT[6] = 0.	LUT[7] = 0.	LUT[8] = 1.	Bleu

L'utilisateur peut modifier ces valeurs enregistrées dans un fichier en modifiant ce même fichier.

Pour chaque coupe : *avec i le numéro de la coupe*
-Calcul de la largeur_moyenne[i] entre chacun des 30 points et le centre de gravité correspondant à la coupe.
-si largeur_moyenne[i]< largeur_moyenne_minimale alors
largeur_moyenne_minimale=largeur_moyenne[i].
-si largeur_moyenne[i]< largeur_moyenne_maximale alors
largeur_moyenne_maximale=largeur_moyenne[i].
-largeur_moyenne_globale += largeur_moyenne[i].
Fin pour

largeur_moyenne_globale /= nb_Coupe.
// Normalisation des valeurs de largeurs moyennes minimale et maximale
largeur_moyenne_minimale = largeur_moyenne_minimale - largeur_moyenne_globale.
largeur_moyenne_maximale = largeur_moyenne_maximale - largeur_moyenne_globale.

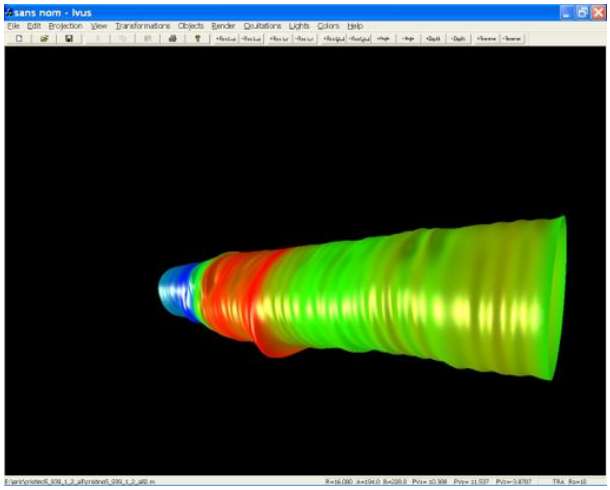
Pour chaque coupe : *avec i le numéro de la coupe*

```
// Normalisation de la valeur de la largeur moyenne
-largeur_moyenne[i] = largeur_moyenne[i] - largeur_moyenne_globale.
-si (largeur_moyenne[i]<0)
        // valeur comprise dans la LUT entre le vert et le bleu
        largeur_moyenne = (largeur_moyenne[i] – largeur_moyenne_minimale)*32 / (0 –
largeur_moyenne_minimale).
        // valeur comprise dans la LUT entre le rouge et le vert
 sinon largeur_moyenne = 32 + (largeur_moyenne[i] – 0)*32 / (largeur_moyenne_maximale - 0).

//Mise a jour de la position correspondante dans la LUT
-positionLUT = largeur_moyenne.

// recherche de la position la plus proche de la position effective « positionLUT » dans la LUT en
 fonction de la résolution de la LUT
-v = 0
-tant que ((v*64/(resolLut-1)) <= positionLUT)
        v++.
-fin tant que

// v = position supérieure la plus proche dans la LUT
// Test la position la plus proche entre la position juste au dessus et celle juste au dessous
-si ( ((v*64/(resolLut-1)) - positionLUT) <  (((v-1)*64/(resolLut-1)) - positionLUT) )
                positionLUT = v*60/(resolLut-1).
-sinon positionLUT = (v-1)*64/(resolLut-1).

// si la position est située entre celle du Bleu et du Vert
-si((positionLUT >=  0)&&( positionLUT <= 32)){
                r = ((positionLUT -0.)*(LUT[3]-LUT[6])/32.)+LUT[6];
                g = ((positionLUT -0.)*(LUT[4]-LUT[7])/32.)+LUT[7];
                b = ((positionLUT -0.)*(LUT[5]-LUT[8])/32.)+LUT[8];
        }
-fin si
// si la position est située entre celle du Bleu et du Vert
-si((positionLUT >  32)&&( positionLUT <= 64)){
                r = ((positionLUT -32.)*(LUT[0]-LUT[3])/32.)+LUT[3];
                g = ((positionLUT -32.)*(LUT[1]-LUT[4])/32.)+LUT[4];
                b = ((positionLUT -32.)*(LUT[2]-LUT[5])/32.)+LUT[5];
        }
-fin si

// Les variables r,g,b correspondent aux composantes de la couleur qui sera assignée à chacun des points
de la coupe i
```

Fin pour

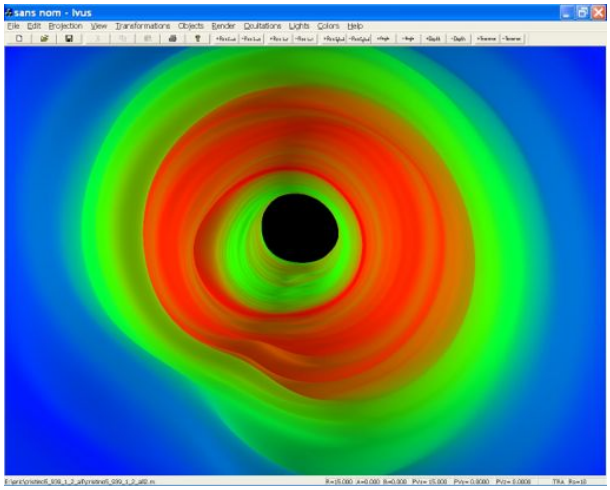
Démonstrations :

Figure 10



Rendu du vaisseau avec mise en évidence de l'élargissement

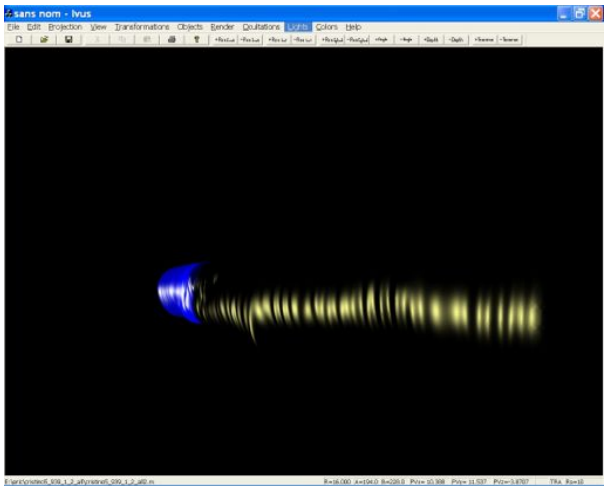
Figure 11



Rendu du vaisseau avec mise en évidence de l'élargissement en mode « parcours »

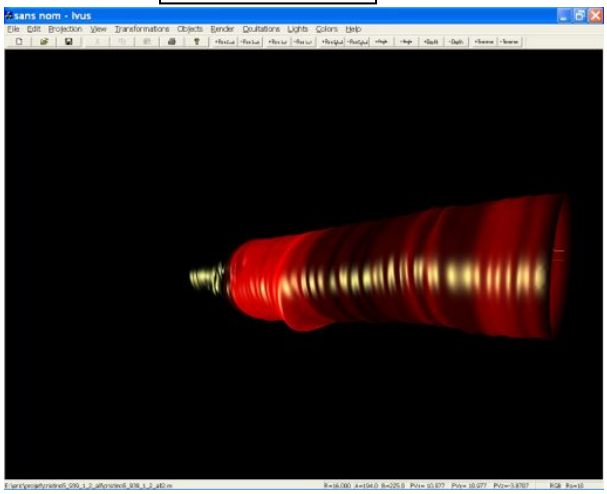
Les figures 10 & 11 correspondent au rendu du vaisseau avec une mise en évidence par une représentation couleur de l'élargissement local de ce dernier en mode perspective et en mode parcours. Nous pouvons ainsi voir en rouge les zones les plus larges, en bleu les zones les plus étroites et en vert les zones moyennes.

Figure 12



Rendu du vaisseau avec mise en évidence de l'élargissement après modification de la LUT

Figure 13



Rendu du vaisseau avec mise en évidence de l'élargissement après modification de la LUT

Les figures 12 & 13 correspondent au rendu du vaisseau avec une mise en évidence par une représentation couleur de l'élargissement local de ce dernier après modification du fichier correspondant à la LUT.

- La LUT correspondant au rendu de la figure 12 est :

Noir (64)
Noir (32)
Bleu (0)

Ainsi apparaissent seulement les zones les plus étroites du vaisseau.

- La LUT correspondant au rendu de la figure 13 est :

Rouge (64)
Noir (32)
Noir (0)

Ainsi apparaissent seulement les zones les plus larges du vaisseau.

Distance par rapport au cathéter ou par rapport au centre gravité

Le principe de cette représentation couleur consiste à définir, pour chaque point composant l'objet,la couleur représentative de la distance par rapport au cathéter ou par rapport au centre de gravité,selon le choix de l'utilisateur,et assigner cette couleur au point. Cette représentation à pour but de représenter les variations de distance entre la surface du vaisseau et le cathéter (ou le centre de gravité), ainsi les irrégularités de la parois du vaisseau seront mises en évidence, par le biais d'une couleur représentative.

L'algorithme de ce calcul est :

Variables utilisées :

nb_Pts_Coupe = entier égal au nombre de points composant une coupe.
nb_Coupe = entier égal au nombre de coupes composant le vaisseau.
distance[i] = distance euclidienne pour une coupe,i, entre les points de la coupe et le centre de gravité (resp. cathéter).
distance_moyenne = valeur moyenne des moyennes de chaque coupe
distance_minimale = distance minimale entre un point de la surface du vaisseau et le centre de gravité courant.
distance_maximale = distance maximale entre un point de la surface du vaisseau et le centre de gravité courant.
resol_LUT = résolution de la LUT (nombre maximum de couleur définissant la LUT)
positionLUT = position de la couleur courante dans la LUT
LUT = matrice 3*3 contenant la couleur des trois paliers définissant la base de la LUT

Par défaut :

LUT[0] = 1.	LUT[1] = 0.	LUT[2] = 0.	Rouge
LUT[3] = 0.	LUT[4] = 1.	LUT[5] = 0.	Vert
LUT[6] = 0.	LUT[7] = 0.	LUT[8] = 1.	Bleu

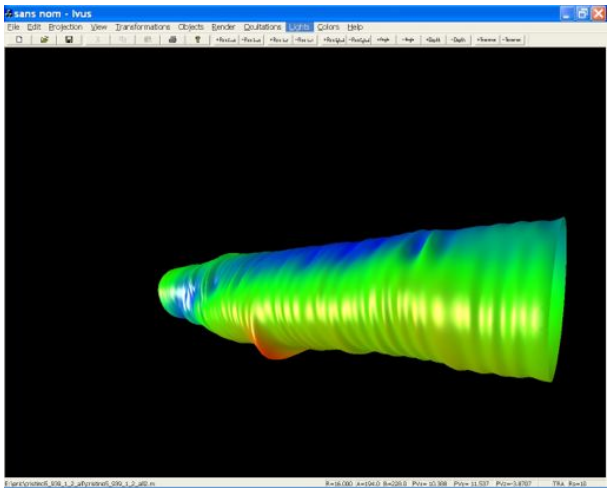
L'utilisateur peut modifier ces valeurs enregistrées dans un fichier en modifiant ce même fichier.

Pour chaque coupe : *avec i le numéro de la coupe*
  Pour chaque point de la coupe : *avec u le numéro du point*
    -Calcul de la distance[u+i* nb_Pts_Coupe] entre le point[u+i* nb_Pts_Coupe] et le centre_de_gravité[i] (resp. cathéter[i]).
    -si distance[u+i* nb_Pts_Coupe] < distance_minimale
      alors distance_minimale= distance[u+i* nb_Pts_Coupe].
    -si distance[u+i* nb_Pts_Coupe] < distance_maximale

```
                    alors distance_maximale= distance[u+i* nb_Pts_Coupe].
                -distance_moyenne += distance[u+i* nb_Pts_Coupe].
        Fin pour
Fin pour

distance_moyenne /= nb_Coupe * nb_Pts_Coupe.
// normalisation des valeurs de distance minimale et maximale
distance_minimale = distance_minimale - distance_moyenne.
distance_maximale = distance_maximale - distance_moyenne.

Pour chaque coupe : avec i le numéro de la coupe
    Pour chaque point de la coupe : avec u le numéro du point
        // normalisation de la distance
        - distance[u+i* nb_Pts_Coupe] = distance[u+i* nb_Pts_Coupe] - distance_moyenne.
        -si (distance[u+i* nb_Pts_Coupe] < 0)
        // valeur comprise dans la LUT entre le vert et le bleu
                alors distance[u+i* nb_Pts_Coupe] = (distance[u+i* nb_Pts_Coupe] –
                    distance_minimale)*32 / (0 – distance_minimale).
        // valeur comprise dans la LUT entre le rouge et le vert
         sinon distance[u+i* nb_Pts_Coupe] = 32 + (distance[u+i* nb_Pts_Coupe] – 0)*32 /
                (distance_maximale - 0).

        //Mise a jour de la position correspondante dans la LUT
        -positionLUT = distance[u+i* nb_Pts_Coupe].

        // recherche de la position la plus proche de la position réelle « positionLUT » dans la LUT en
         fonction de la résolution de la LUT
        -v = 0
        -tant que ((v*64/(resolLut-1)) <= positionLUT)
                v++.
        -fin tant que

        // v = position supérieure la plus proche dans la LUT
        // Test la position la plus proche entre la position juste au dessus et celle juste au dessous
        -si ( ((v*64/(resolLut-1)) - positionLUT) <  (((v-1)*64/(resolLut-1)) - positionLUT) )
                        positionLUT = v*60/(resolLut-1).
        -sinon positionLUT = (v-1)*64/(resolLut-1).

        // si la position est située entre celle du Bleu et du Vert
        -si((positionLUT >=  0)&&( positionLUT <= 32)){
                r = ((positionLUT -0.)*(LUT[3]-LUT[6])/32.)+LUT[6];
                g = ((positionLUT -0.)*(LUT[4]-LUT[7])/32.)+LUT[7];
                b = ((positionLUT -0.)*(LUT[5]-LUT[8])/32.)+LUT[8];
                }
        -fin si
        // si la position est située entre celle du Bleu et du Vert
        -si((positionLUT >  32)&&( positionLUT <= 64)){
                r = ((positionLUT -32.)*(LUT[0]-LUT[3])/32.)+LUT[3];
                g = ((positionLUT -32.)*(LUT[1]-LUT[4])/32.)+LUT[4];
                b = ((positionLUT -32.)*(LUT[2]-LUT[5])/32.)+LUT[5];
                }
        -fin si
        // Les variables r,g,b correspondent aux composantes de la couleur qui sera assignée au point
                courant
    Fin pour
Fin pour
```

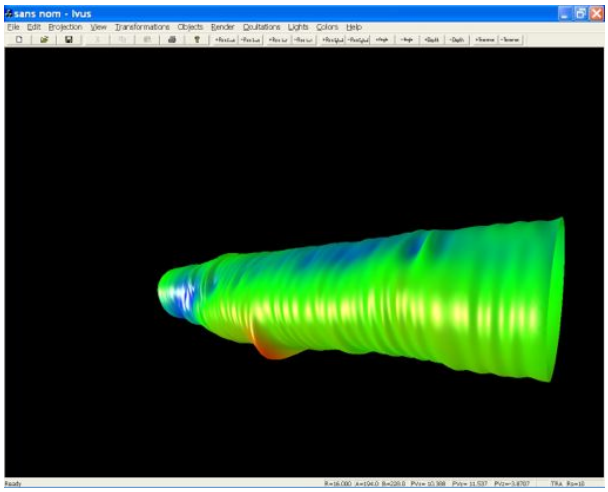
Démonstrations :

Figure 14



Rendu du vaisseau avec mise en évidence des distances entre la surface du vaisseau et le centre de gravité

Figure 15



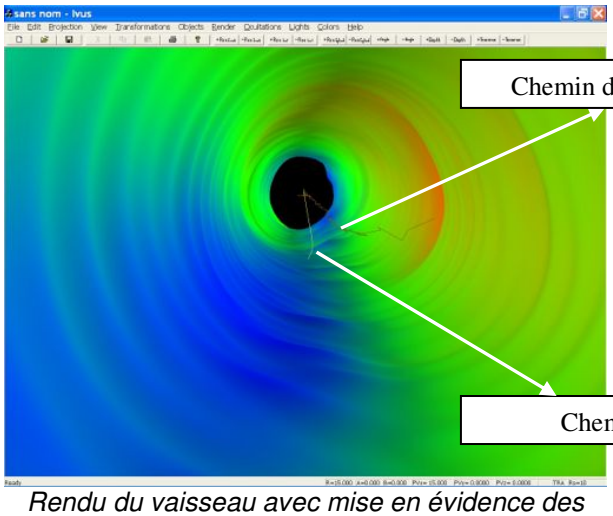
Rendu du vaisseau avec mise en évidence des distances entre la surface du vaisseau et de la position du cathéter

Les figures 14 & 15 correspondent au rendu du vaisseau avec une mise en évidence des distances entre la surface du vaisseau et le centre de gravité (resp. la position du cathéter) par une représentation couleur.

Nous pouvons ainsi voir sur la figure 14, en rouge les zones les plus éloignées de l'axe central du vaisseau,en bleu les zones les plus proches et en vert les zones caractérisants des distances moyennes à l'axe central. En effet, nous supposons que l'axe central du vaisseau peut être défini comme étant l'ensemble des centres de gravité de chacune des coupes.

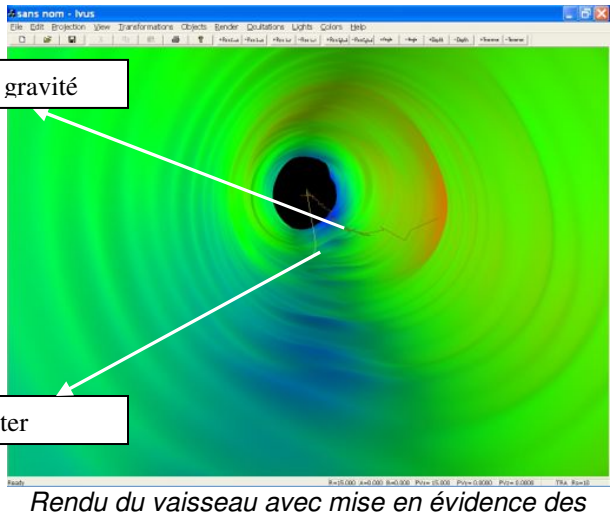
La mise en évidence des distances entre les points composants la surface du vaisseau et les positions du cathéter peuvent permettre d'analyser le trajet du cathéter en cas de problème lors de la propagation de ce dernier à l'intérieur du vaisseau. Nous voyons par exemple que la zone de couleur bleue visible sur la figure 14 paraît moins étendue sur la figure 15 ce qui nous laisse penser que le cathéter a dévié de l'axe central du vaisseau lors de sa propagation.

Figure 16



Rendu du vaisseau avec mise en évidence des distances entre la surface du vaisseau et du centre de gravité en mode « parcours »

Figure 17



Rendu du vaisseau avec mise en évidence des distances entre la surface du vaisseau et de la position du cathéter en mode « parcours »

Les figures 16 & 17 correspondent au rendu du vaisseau avec une mise en évidence, par une représentation couleur,des distances entre la surface du vaisseau et le centre de gravité (resp. la position du cathéter) en mode « parcours ».Ces deux visualisations permettent par exemple de voir la zone bleu apparaissant en premier plan sur la figure 16 est moins importante sur la figure 17 ce qui signifie que la propagation du cathéter à dévié de l'axe central du vaisseau.

4. Problème(s) rencontré(s)

Le premier problème fut d'appliquer les couleurs sur la NURBS. Lors de la création d'une surface NURBS, nous ne manipulons pas de Vertex. La création de ces Vertex se fait par l'intermédiaire d'une fonction gluNurbsSurface proposée par la librarie GLU . C'est pourquoi je ne pouvais pas appliquer directement une couleur sur ces points 3D n'étant pas accessibles directement. Mon premier choix fut de constituer une structure image de type BMP et de texturer cette image sur la NURBS. La création et la manipulation d'une telle structure étant très lourde,j'ai décidé d'utiliser une structure dynamique (de type PILE) dans laquelle était stockée toutes les valeurs de couleurs correspondant à chaque point définissant la surface de l'objet. Le rappel de la fonction gluNurbsSurface permet d'utiliser directement une structure et d'assigner les couleurs stockées à chacun des points définissant la NURBS.

Le second problème concernait l'utilisation de la LUT. N'ayant pas fixé un nombre limite de couleurs définissants la LUT hormis les trois couleurs références,le nombre de couleurs générées,par le processus d'interpolation,était illimité,les représentations couleurs étaient alors plus lourdes,inutilement,et parfois non homogènes. J'ai donc choisi de fixer une résolution pour la LUT (modifiable par l'utilisateur) et désormais, lorsque l'on cherche une couleur, dans la LUT, correspondant à une valeur,on cherche en fait la couleur la plus proche de la couleur réelle correspondant à cette valeur.

TEXTURE DU VAISSEAU

1. *Problématique*

Le dernier objectif de ce projet concernait la création et le plaquage d'une texture (pouvant être transparente), en 3 dimensions,sur le vaisseau à partir des images IVUS.

Les dernières versions d'OpenGL associées à des cartes graphiques relativement récentes permettent de définir des textures 3D sur des objets. Cependant,le vaisseau rendu est composé d'un très grand nombre de points. De plus,la manipulation de la texture 3D composée de toutes les images IVUS nécessite beaucoup de ressources. La puissance de mon ordinateur ne m'a malheureusement pas permis d'utiliser les primitives OpenGL pour définir et plaquer directement ma texture 3D sur mon objet 3D. C'est pourquoi deux autres méthodes ont été retenues pour réaliser ce travail. C'est deux méthodes sont composées d'opérations manipulant quasiment que des textures 2D tout permettant d'obtenir des résultats analogues à ceux espérés en utilisant les textures 3D d'OpenGL.

La première méthode consiste à construire une succession de plans, caractérisant chacune des coupes du vaisseau, et de texturer sur chacun de ces plans l'image IVUS lui correspondant.

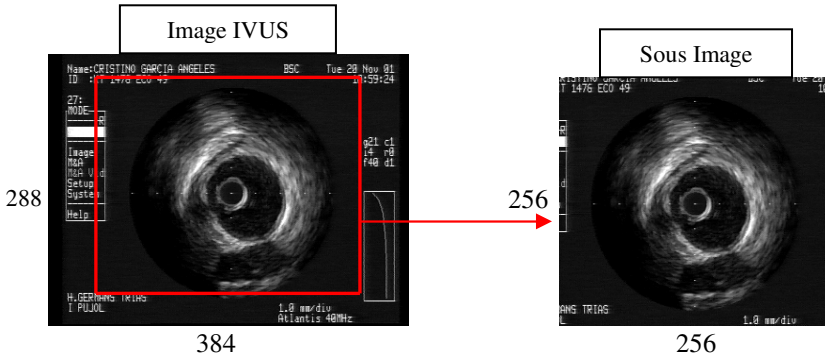
La seconde méthode est basée sur la construction d'un cylindre « creux » (en analogie avec la forme d'un vaisseau) auquel j'ai appliqué sur chaque polygone visible composant ce cylindre une texture 2D.

Dans un premier temps, il a fallu modifier les images IVUS et en éliminer les informations inutiles.

2. *Modification des images IVUS*

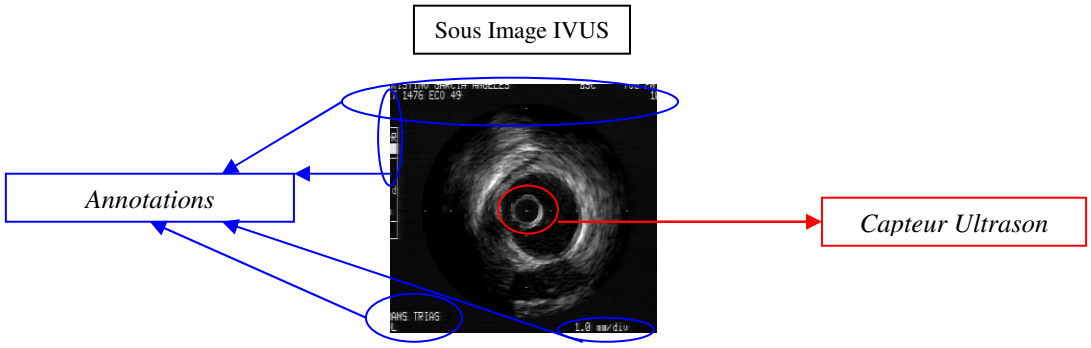
Sous OpenGL, pour pouvoir définir un Objet-Texture à partir d'une image,il faut que les dimensions (a,b) de cette image soit tel que a = 2^x et b = 2^y ,avec x et y Є N. Hors, les dimensions des images IVUS sont de 384 * 288 . Il fut donc nécessaire de redimensionner ces images.

La solution choisie a été, pour une image IVUS donnée,de couper cette image afin de récupérer sa sous image centrée de dimension 256 * 256.

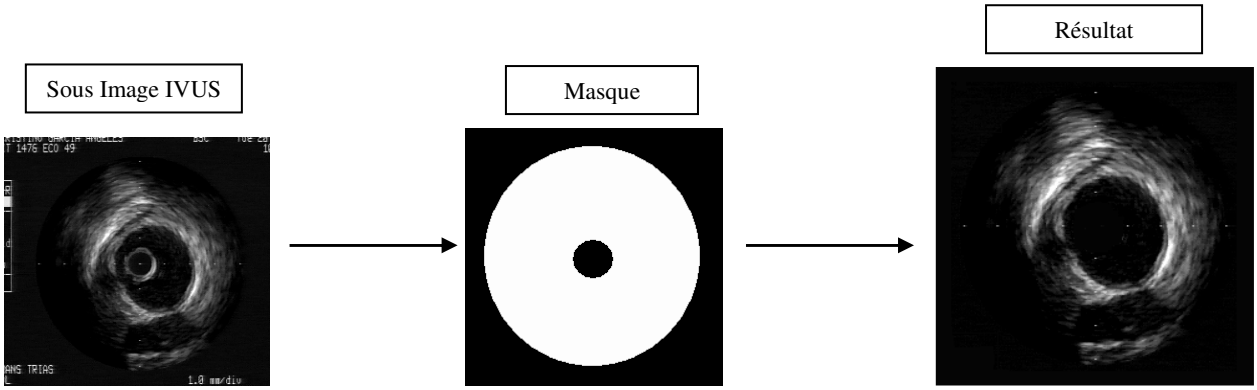


Sur les images IVUS,les informations que nous souhaitions visualiser correspondent aux structures les plus denses du vaisseau,l' Adventitia ou encore les plaques de calcium (cf. INTRODUCTION).C'est pourquoi nous avons décidé d'intégrer à notre projet un mode « transparence » afin de mettre en évidence les zones très claires (très denses) de l'images IVUS.

Cependant, certaines zones de couleurs blanches composants les images IVUS sont inutiles. En effet, il y a des annotations,dont les caractères sont blancs,sur les images IVUS et elles génèrent du bruit lors d'une visualisation en mode « transparence ».Ces annotations étant situées sur les bords des images IVUS,j'ai décidé de créer un masque afin de mettre en noir les pixels situés hors d'un certains cercle (cercle centré sur l'image) et d'appliquer ce masque sur mes images. Le petit cercle central représenté sur les images IVUS ne constitue pas une information pertinente puisqu'il correspond au capteur ultrason composant le cathéter. Aussi j'ai décidé de rajouter sur mon masque un petit cercle centré, de la dimension du capteur composant le cathéter.



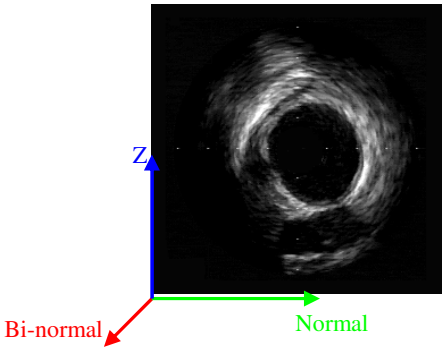
Processus :



3. *Texture 3D par plans du vaisseau*

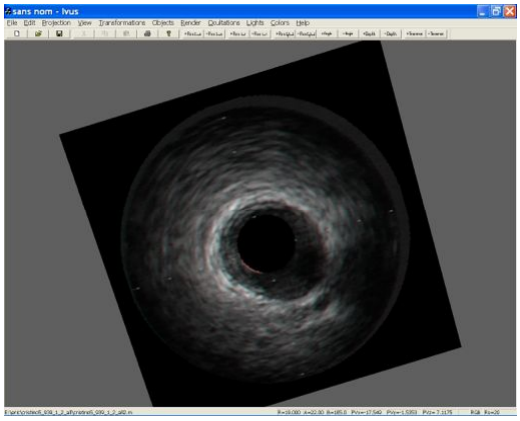
Cette méthode consiste donc à construire une succession de plans, centrés sur chacune des positions du cathéter. Chacun de ces plans représentera alors une des coupes. Sur un plan donné, l'image IVUS correspondante y sera alors plaquée.

Mon premier travail a consisté à modéliser tous les plans le long du chemin du cathéter. Durant sa propagation à l'intérieur du vaisseau, le cathéter a pu pivoter sur lui-même et ainsi modifier son orientation. C'est pourquoi j'ai prêté une attention particulière à l'orientation de chacun des plans. Ainsi, j'ai récupéré auprès de l'équipe,les informations concernant l'orientation de chacune des images IVUS (vecteur normal,vecteur bi-normal).



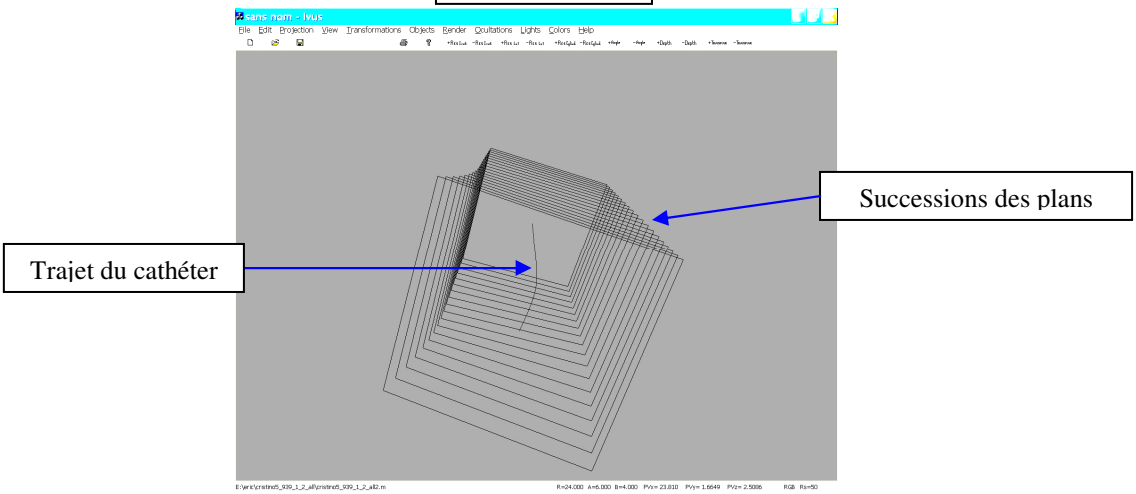
Par exemple, la visualisation de la première image à l'issue des opérations d'orientations est :

Rendu de la 1ere image



J'ai donc construit la succession des plans, centrés sur chacune des positions du cathéter composant son parcours.

Figure 17

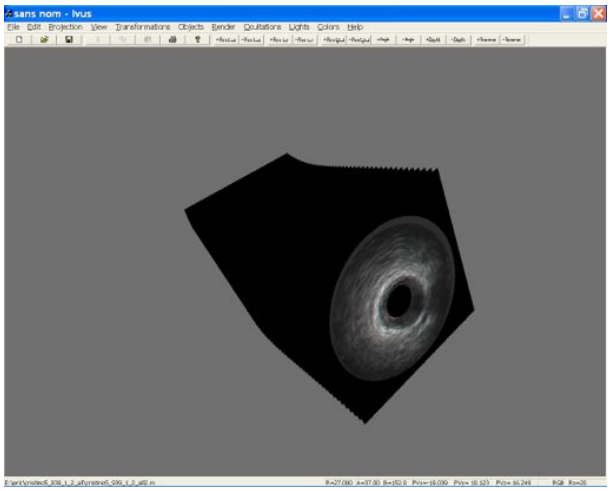


Rendu filaire de la succession des plans

Sur la figure 17, nous pouvons voir le rendu filaire de la succession de plans le long de trajet réalisé par le cathéter. En effet, il apparaît sur cette figure les plans caractérisant chacune des coupes ainsi que le trajet du cathéter au centre de la fenêtre de visualisation.

La dernière étape de cette méthode consistait à spécifier et plaquer les textures sur chaque plan modélisé et de définir un effet de transparence. L'utilisation de primitives fournies par OpenGL a permis de définir et assigner les textures.

Figure 17



Rendu des textures planaires sans transparence

Sur la figure 17, nous pouvons voir le rendu des textures planaires sans transparence.

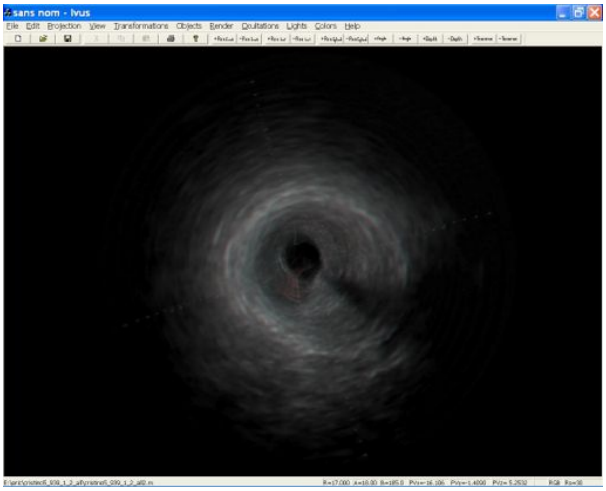
Pour simuler le phénomène de transparence souhaité, j'ai utilisé le « blending » définit dans OpenGL. Le procédé de « blending » est utilisé pour réaliser un mélange de couleurs. Au cours du « blending », un coefficient appelé facteur de blending source est appliqué à la valeur de la source (le fragment en cour de traitement), et un coefficient appelé facteur de blending destination est appliqué à la valeur de la destination (le fragment supposé être caché derrière le fragment en cour de traitement). En définissant la valeur du facteur de blending source comme étant égale à la valeur d'intensité du fragment en cour de traitement et en définissant la valeur du facteur de blending destination comme étant égale à 1 – intensité du fragment de destination, j'ai défini un mélange de couleurs (resp. niveau de gris) comparable à un effet de transparence.

Rmq : l'intensité d'un pixel est comprise entre 0 et 1.

De même, il était intéressant de pouvoir filtrer les fragments selon leur intensité. Par exemple, faire seulement apparaître les fragments ayant une intensité très importante peut permettre la mise en évidence de plaque de calcium (dont l'intensité est très élevé).C'est pourquoi j'ai utilisé l' « Alpha Test » implémenté par OpenGL et j'ai défini le seuil sur Alpha utilisé par ce test, permettant de faire apparaître les fragments selon leur intensité. Aussi, je propose à l'utilisateur de modifier le seuil sur Alpha afin de rendre des fragments plus ou moins intenses.

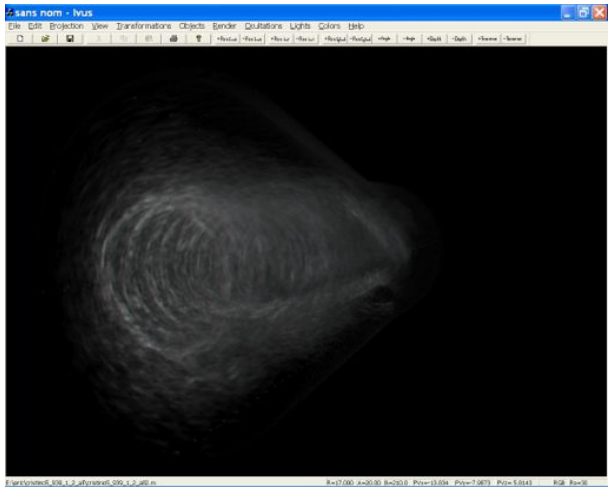
Démonstration :

Figure 18



Rendu des textures planaires avec transparence avec seuil Alpha = 0.1

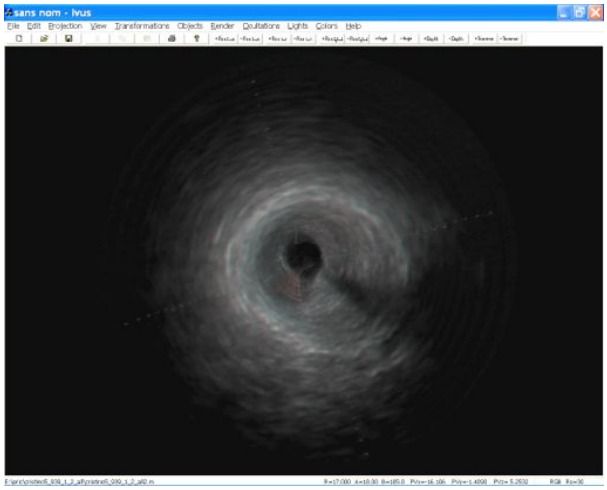
Figure 19



Rendu des textures planaires avec transparence avec seuil Alpha = 0.1

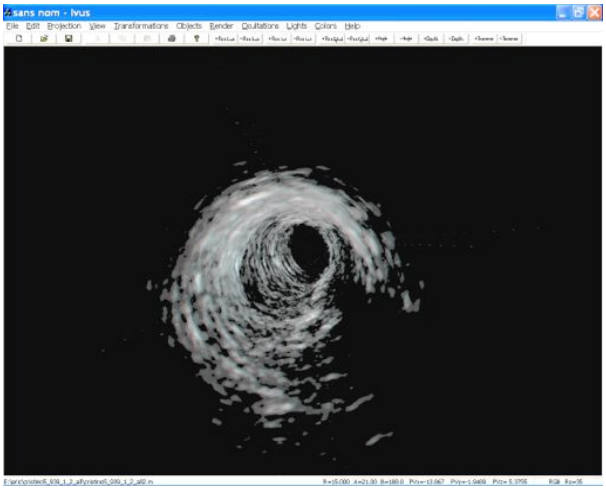
Les figures 18 et 19 correspondent aux rendus des textures planaires avec effet de transparence. Le seuil d'intensité Alpha est égal à 0,1 pour les deux rendus. La figure 18 présente un rendu des textures planaires de face. La figure 19 présente un rendu des textures planaires visualisées légèrement de coté.

Figure 20



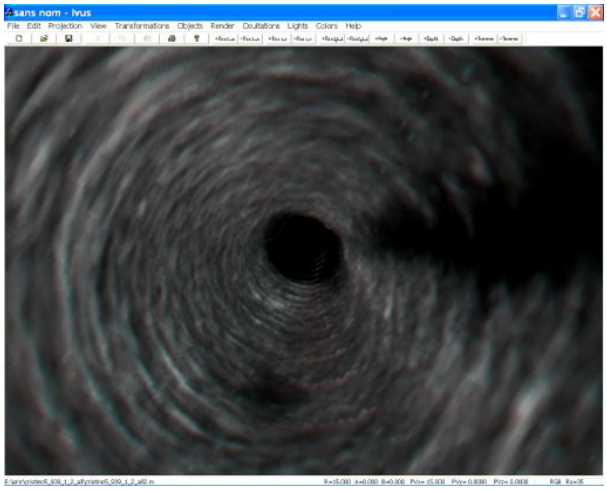
Rendu des textures planaires avec transparence avec seuil Alpha = 0.1

Figure 21



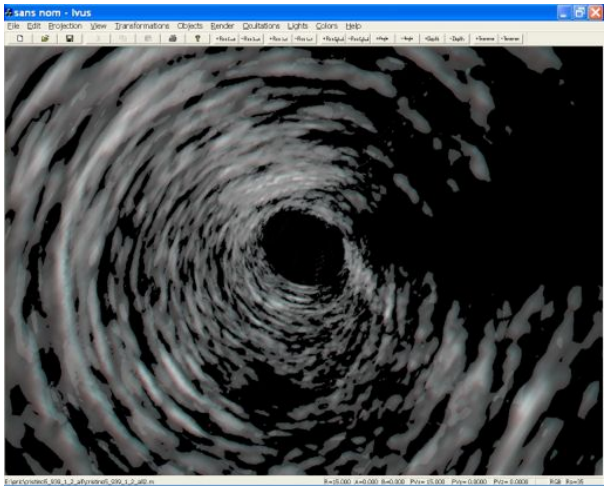
Rendu des textures planaires avec transparence avec seuil Alpha = 0.6

Figure 22



Rendu des textures planaires avec transparence avec seuil Alpha = 0.1 en mode « parcours »

Figure 23



Rendu des textures planaires avec transparence avec seuil Alpha = 0.6 en mode « parcours »

Les figures 20 à 23 représentent les rendus de textures par plans avec effet de transparence, et présentent des jeux de tests sur les valeurs du seuil Alpha. Les figures 22 et 23 présentent le rendu en mode « parcours ».

4. *Texture 3D cylindrique du vaisseau*

Cette méthode est basée sur la construction d'un cylindre « creux » (en analogie avec la forme d'un vaisseau) auquel j'ai appliqué une texture. La méthode précédente permettait d'obtenir un rendu de texture au niveau de chaque coupe, cependant, il n'y avait pas d'interpolations couleurs entre deux plans correspondant à deux coupes.

Nous avons défini cette méthode qui permet de modéliser un objet 3D dont la forme générale est semblable à celle d'un vaisseau. De plus cette méthode permet d'obtenir une réelle texture 3D volumique puisque le volume compris entre deux coupes (deux images IVUS) est texturé par interpolation (ce que la méthode précédente ne réalisait pas). L'inconvénient de cette méthode réside dans la lourdeur de son traitement.

Rmq : les images qui seront utilisées pour spécifier les textures à plaquer, auront elles aussi subies les modifications vu dans le chapitre précédent.

La visualisation correspondant à cette méthode permet de voir l'Adventitia (la chair du vaisseau), puisque le rendu correspond en réalité à une coupe de cylindre. L'utilisateur peut modifier l'angle maximal de construction du cylindre. Par exemple, si cet angle est égal à Pi, l'objet rendu correspondra à la moitié du cylindre (coupe longitudinale) (voir figure 25).Si cet angle est égal à Pi / 2, l'objet rendu correspondra au quart du cylindre (en coupe longitudinale). Par défaut, l'angle de construction du cylindre a été fixé à 4Pi / 3, ainsi les 2 / 3 du cylindre sont visualisés.

Pour constituer mon cylindre représentant le vaisseau, mon choix fut de créer un cylindre entre chaque coupe. L'ensemble de tous les cylindres construits constituera un seul et même cylindre représentant le vaisseau. Le problème résidait alors dans la construction d'un cylindre liant deux coupes.

Pour modéliser un cylindre liant deux coupes, j'ai supposé que cet objet était composé de deux polygones « bases »et de deux surfaces latérales liant ces deux bases. Sachant que la première base dessinée correspond à la coupe courante du vaisseau tandis que la seconde base correspond à la coupe suivante. Les surfaces latérales du cylindre permettent de lier les deux bases donc les deux coupes.

Figure 24

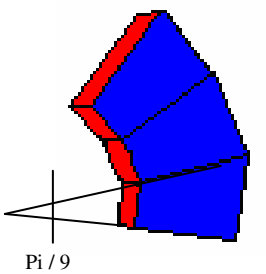
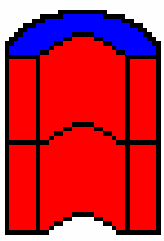


Figure 25



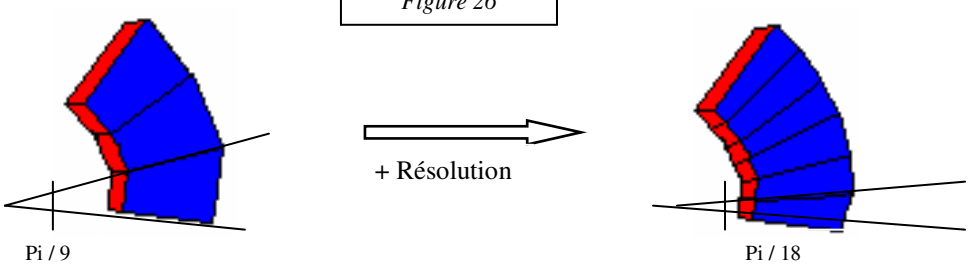
La figure 24 présente un schéma d'un tiers de cylindre vu du dessus. La figure 25 présente un schéma d'une coupe de cylindre dans le sens de la longueur. En bleu sont représentées les bases du cylindre tandis qu'en rouge sont représentées les surfaces latérales du cylindre liant ces deux bases.

Pour modéliser chaque élément (bases et surfaces latérales) d'un cylindre,j'ai choisi de les décomposer en plusieurs quadrilatères afin de faciliter la modélisation.

Lors de la modélisation d'une base, j'ai choisi de dessiner un quadrilatère tout les béta (égal à Pi / 9 par défaut) (voir figure 24) jusqu'à ce que l'on atteigne l'angle maximal de construction (égal à 4Pi / 3 par défaut).J' ai fait de même concernant la construction des surfaces latérales. Je dessine un quadrilatère joignant les deux bases du cylindre tout les béta jusqu'à que l'on l'angle maximal de construction.

Je propose à l'utilisateur de modifier la résolution du cylindre afin qu'il puisse adapter le rendu à la puissance de son ordinateur. En effet, j'augmente le nombre de points définissant mon objet en réduisant la valeur de mon angle béta. Réduire la valeur de béta augmente le nombre de quadrilatères définissant l'objet et donc augmente aussi le nombre de points.

Figure 26



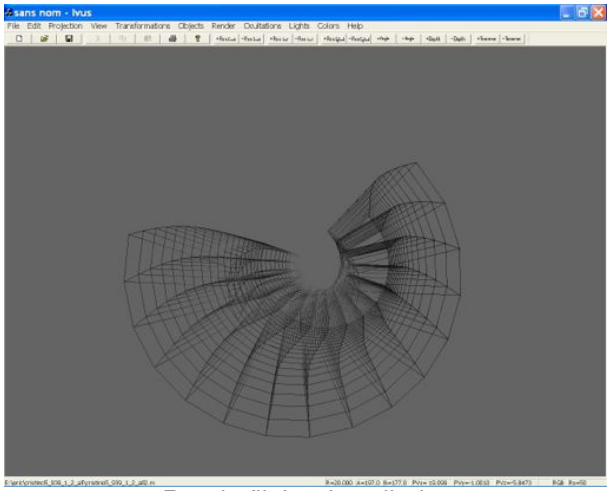
Angle de construction = Pi /3
Béta = Pi / 9
Nb points = 16

Angle de construction = Pi /3
Béta = Pi / 18
Nb Points = 28

La figure 26 présente deux schémas permettant de rendre compte de la technique utilisée pour modifier la résolution du cylindre. En effet, les deux schémas présentent un cylindre dont l'angle de construction est égal à Pi / 3, l'angle béta est plus important sur le cylindre de droite. Ainsi, le cylindre de droite a une meilleure résolution (28 points contre 16 points pour le cylindre de gauche).

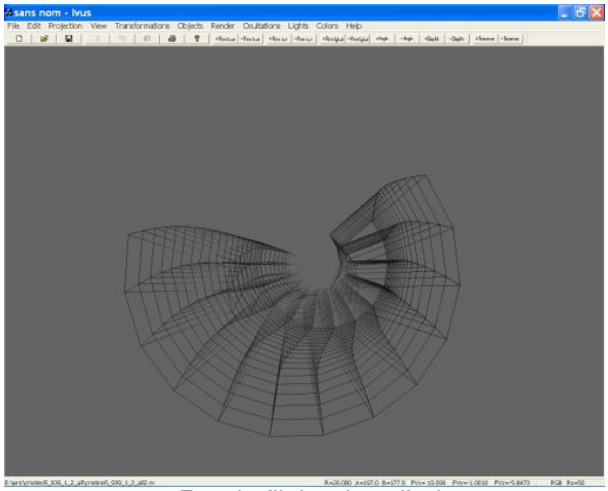
Visualisation de l'objet modélisé :

Figure 27



Rendu filaire du cylindre,
Angle de construction = 4Pi / 3,
Béta = Pi / 11

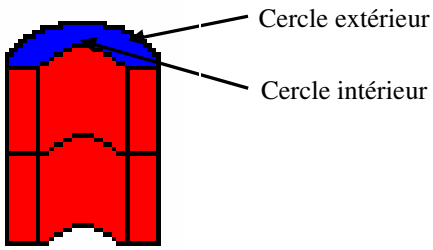
Figure 28



Rendu filaire du cylindre,
Angle de construction = 4Pi / 3
Béta = Pi / 9

Les figures 27 et 28 correspondent aux visualisations du cylindre en mode filaire avec deux résolutions différentes.

Je propose aussi à l'utilisateur de « creuser » plus ou moins le cylindre. Les bases du cylindre peuvent être comparées à des disques creusés étant délimités par un cercle intérieur et un cercle extérieur.



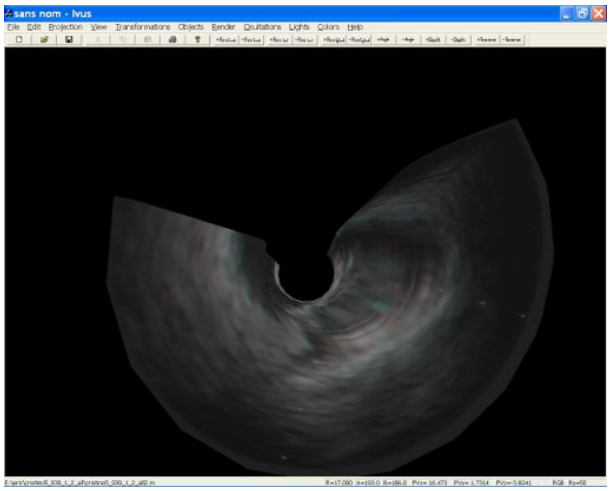
En redéfinissant le rayon du cercle intérieur je « creuse »,plus ou moins,le cylindre ce qui permet de visualiser,plus ou moins,en profondeur de l'Adventitia du vaisseau.

La dernière étape consistait à texturer le cylindre. Plaquer une texture sur le cylindre correspondait à texturer chaque quadrilatère le définissant. Pour chacun des points définissant un quadrilatère, j'ai assigné une texture. Si parmi les quatre points définissant un quadrilatère, deux points ont été texturé à partir de deux images

différentes, alors la texture assignée à ce quadrilatère sera le résultat d'une interpolation entre les couleurs assignées aux quatre points le définissant (ce cas fut rencontré lors de la spécification de la texture des quadrilatères composants les surfaces latérales des sous cylindres).

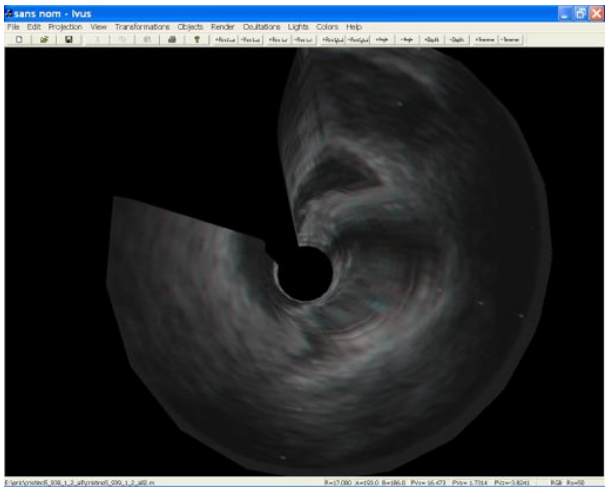
Démonstrations :

Figure 29



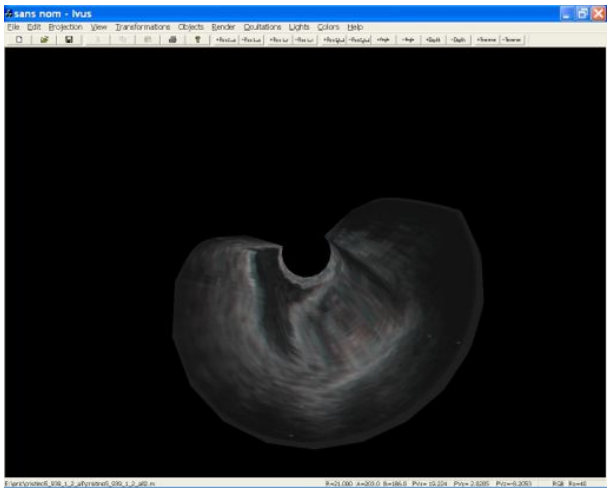
Rendu de la texture cylindrique,
Angle de construction = 4Pi / 3

Figure 30



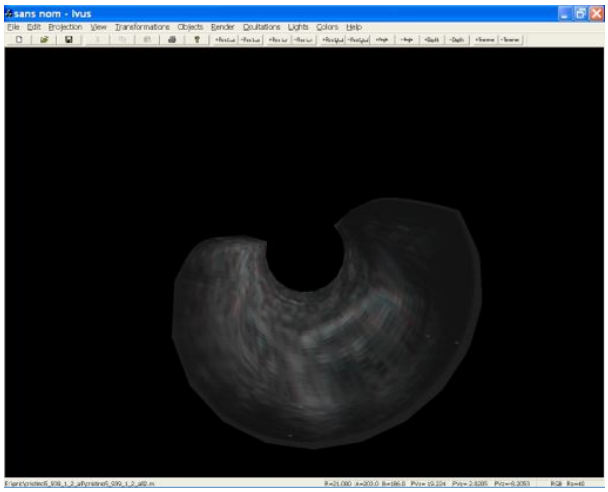
Rendu de la texture cylindrique,
Angle de construction = 5Pi / 3

Figure 31



Rendu de la texture cylindrique creusée,
Angle de construction = 4Pi / 3

Figure 32

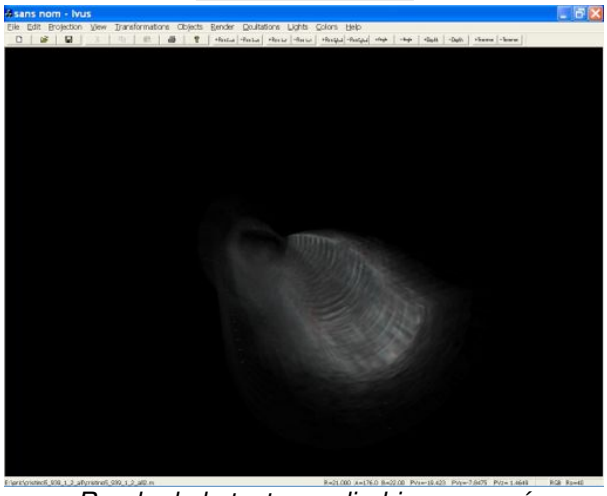


Rendu de la texture cylindrique creusée,
Angle de construction = 4Pi / 3

Les figures 29 et 30 correspondent à la visualisation des rendus de la texture cylindrique avec un angle de construction différent.

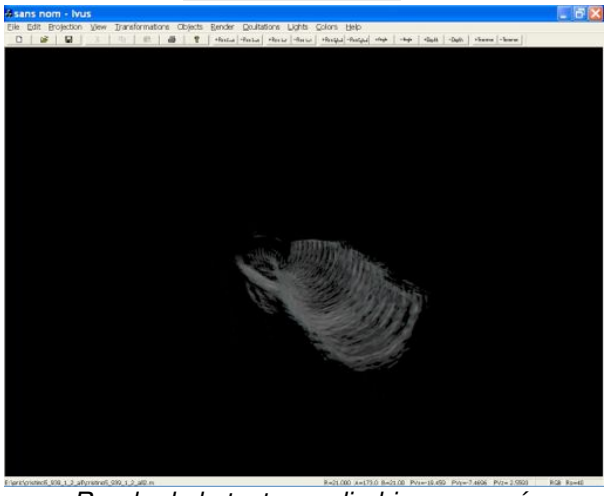
Les figures 31 et 32 correspondent à la visualisation des rendus de la texture cylindrique « creusée ». La figure 32 présente un cylindre plus creusé, ainsi nous pouvons voir au cœur de la « chair » (l' Adventitia) du vaisseau.

Figure 33



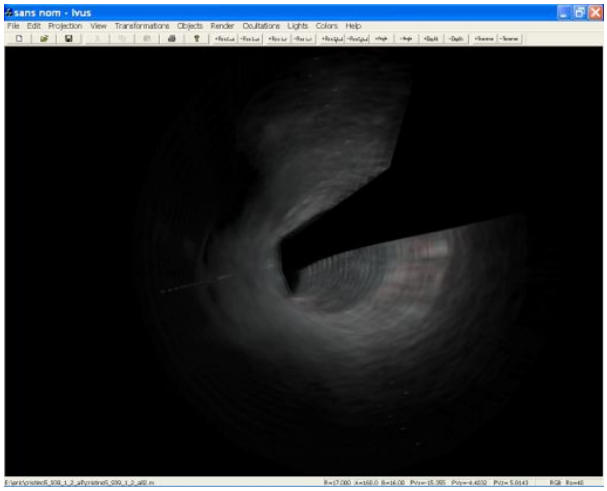
Rendu de la texture cylindrique creusée,
avec effet de transparence,
seuil Alpha = 0,1

Figure 34



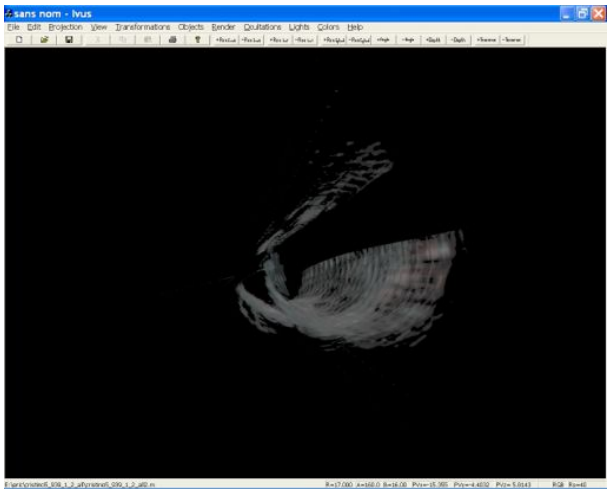
Rendu de la texture cylindrique creusée,
avec effet de transparence,
seuil Alpha = 0,6

Figure 35



Rendu de la texture cylindrique creusée,
Angle de construction = 5Pi / 3
avec effet de transparence,
seuil Alpha = 0,1

Figure 36



Rendu de la texture cylindrique creusée,
Angle de construction = 5Pi / 3
avec effet de transparence,
seuil Alpha = 0,6

Les figures 33 et 34 correspondent à la visualisation des rendus de la texture cylindrique avec un effet de transparence. La valeur du seuil Alpha est plus grande sur la figure 34, les structures les plus claires sont alors mises en évidence.

Les figures 35 et 36 sont similaires aux figures 33 et 34 hormis l'angle de construction qui est plus important.

5. Problème(s) rencontré(s)

Le problème majeur relatif à l'implémentation de cette visualisation concernait la spécification des textures appliquées aux points composants les surfaces latérales des sous cylindres. Une extrémité d'un quadrilatère, définissant une surface latérale, correspondait à une image tandis qu'à l'autre extrémité correspondait une autre image. Lorsque l'on souhaite plaquer une texture sur un objet, sous OpenGL, il faut spécifier la texture à lier à l'objet en même temps que l'on modélise cet objet. Ainsi ces quadrilatères devaient être liés à deux images (soit deux textures 2D) ce qui est impossible en utilisant le procédé de Texture 2D. Nous avons donc choisi d'utiliser les Textures 3D contenant les deux images à lier à l'objet.

BILAN DU STAGE

Ce stage fut très enrichissant d'un point de vue humain et technique. Le projet m'a énormément intéressé. Les objectifs du cahier des charges ont été réalisés. Il serait, cependant, intéressant de poursuivre ces travaux et améliorer par exemple le rendu de texture en utilisant seulement le procédé de Texture 3D (la puissance de mon matériel de travail ne me l'a pas permis). L'intégration d'un module d'interaction Homme/Machine permettant la manipulation d'objet dans un espace à trois dimensions (à l'aide d'un gant par exemple) pourrait être adapté à cette application.

REMERCIEMENT

Je souhaite adresser un remerciement particulier à mon responsable de stage au sein du CVC, Mr Enric Martì, qui m'a aidé tout au long du projet.

REFERENCES

OpenGL Reference Manual
Second Edition
The official Reference Document
to OpenGL, Version 1.1

OpenGL Architechture Review Bord
Editors : Renate Kempf and Chris Frazier

OpenGL Programming Guide
Fourth Edition
The Official Guide to Learning
OpenGL, Version 1.4

OpenGL Architechture Review Bord
Editors : Dave Shreiner, Mason Woo, Jackie Neider, Tom Davis

The OpenGL EXTENSION GUIDE

Editor : Eric Lengyel